A Node Based Strategy to Integrate Web Based Resources

Brad Marshall, BDGP, BioXML.org

Abstract:

The rapid growth of web based biology resources threatens to overwhelm even the most web–savvy biologist. Even though many of these tools are excellent, there are no standard ways to locate these resources, use their interfaces or input and export data. Additionally, there are too many resources, changing too fast for one organization or group of organizations to integrate efficiently into a data warehouse or other higher level organizational scheme. A proper solution to the problem must be one that is distributed and in which parties providing web–based services can join of their own volition. I am proposing a simple, two–pronged strategy to provide a low level integration of web based services which requires the use of XML as the data transport layer. The first part of the strategy is to provide guidelines for the resources themselves. This includes a specification for how an individual resource can input and export data, as well as how it can be located and queried. Any resource which adheres to these guidelines will become a "node" of the system. The second part of the strategy involves creating a metadata standard to describe the nodes. For this I am proposing a standard set of extensions to the w3c's Resource Definition Framework.

Node–World:

The central concept behind this proposal is that of viewing a web url as a "node" of a larger system. A set of guidelines will be established and any web resource which adheres to the guidelines will count as a node. Guidelines will include but not be limited to:

1) Input and export XML formats.
2) How data is obtained (push vs. pull)
3) How a url can be queried.
4) How additional arguments should be specified.

The two most low–level node types are data sources and general applications. An example data source is Genbank. It is queried simply and returns a sequence. To count as a node, however, it would have to export it's data in an XML format. A third party could build a proxy node which queried Genbank and returned the data in an XML format. An example application is Blast. It accepts data (a sequence), does an analysis and returns a result. Again, to count as a node a blast server would have to import and export XML. More node types can be built as specializations of these "base class" node types. Examples follow:

Specialized Application Node Types:

1) Data Format Converters
   – use XSL to convert one XML format to another
2) Locator Nodes
   – a node which indexes other nodes
   – can be queried by input format, export format and application
3) Display Nodes

– use XSL to convert XML into browser viewable HTML

Additionally, these low–level nodes can be combined to create high level nodes. Examples could include:

1) Meta–Locator Nodes
   – these could query other locator nodes
2) Data Warehouse Style Nodes
   – query many Data Source nodes

Metadata:

The remaining piece of the puzzle is a way to describe the nodes of the system. I propose to create an extension to the Resource Definition Framework (RDF). The idea is similar to that of the Dublin Core, which is a standard RDF extension aimed at describing general web content. RDF is a series of object, subject, predicate triples which allows one to give attributes to web resources. An attribute can itself be an RDF triple, allowing one to build arbitrarily large trees of resources. All objects, subjects and predicates are identified by a fully formed uri. Once an RDF document is in place it could be indexed by a locator node.

Things that need to be specified by the extension are:

1) Node locations.
2) Node types.
3) Node input/export formats.
4) How a node is queried.
5) Data Retrieval – push vs. pull.
6) Additional Node Parameters

Conclusions:

I am proposing a simple framework to allow low level integration of web based resources. The hope is that this is a reasonably easily implemented standard to allow both human and computer based clients to begin to integrate the many services available on the web. In order to develop for the system, a resource provider simply has to make their url read and/or write XML, put up an RDF document and make that document available to an existing locator node. Since the system is distributed, it should scale in a manner similar to the current web, but with more integration. The biggest roadblock to this, or any similar system, is simply in gaining a critical mass of participating sites.